FUZZY SYSTEMS:

Basics using *MATLAB Fuzzy Toolbox*

By

Heikki N. Koivo

2.1. Fuzzy sets

Membership functions

Universal discourse U set of elements, $\{u\}$.

Fuzzy set F in universal discourse U:

Membership function μ_F *(membership function)*.

$$\mu_F : U \rightarrow [0,1]$$

Fuzzy set

$$F = \{x, \mu_F(x) | x \in U\}$$

The value of the membership function μ_F(u) describes the degree of membership of u in the fuzzy set F. It takes values between 0 and 1.

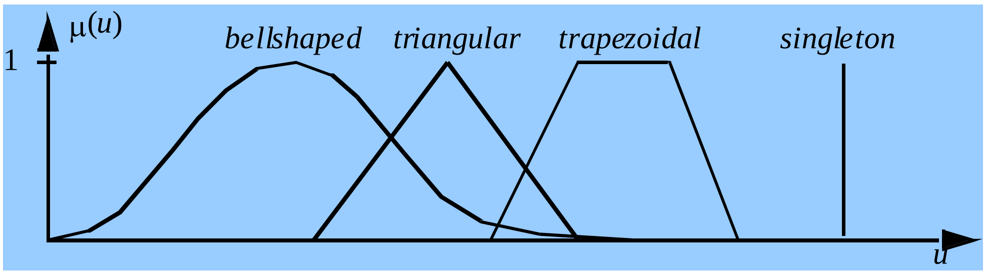


Fig.2. 1. Membership functions may assume different shapes like bell-shaped, triangular, trapezoidal and singleton.

2. 2. USING MATLAB Fuzzy Toolbox GUI

PROBLEM 2.1. Let the room temperature **T** be a fuzzy variable. Characterize it with three different (fuzzy) temperatures: *cold,warm, hot*.

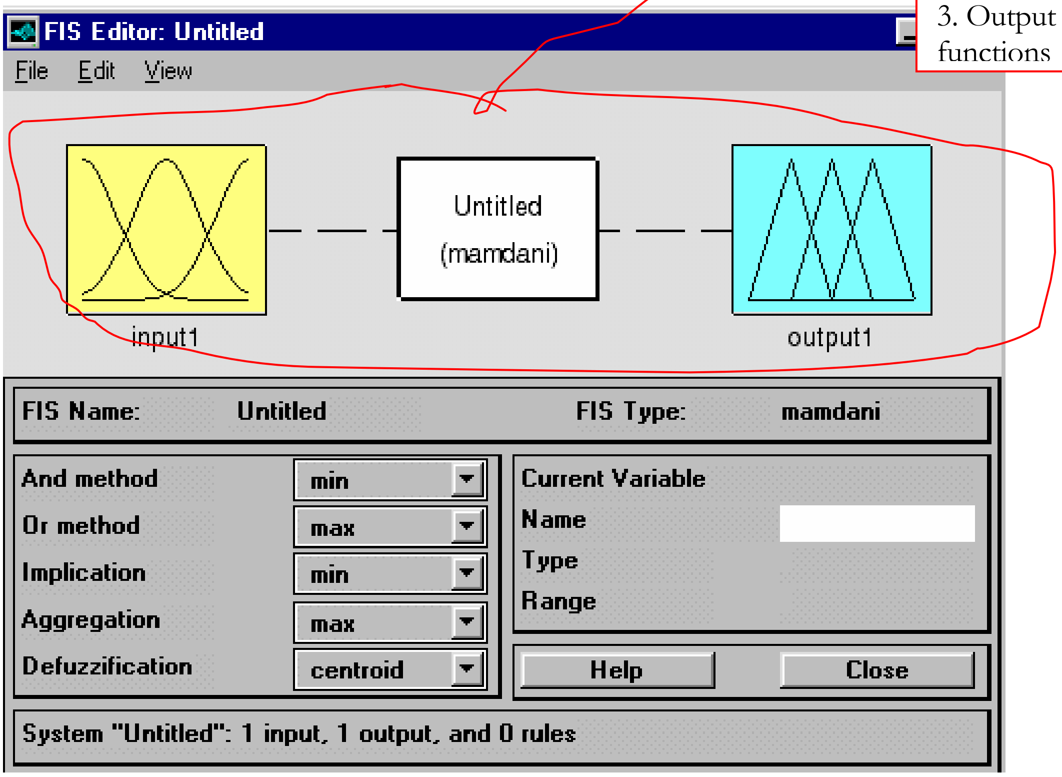
SOLUTION:

1. First define the temperature range, e.g. [0^0,40^0].

2. When MATLAB is open, then open *GUI* (*GUI = Graphical User Interface*) by typing ***fuzzy***

The result is shown below:

1. Input membership functions
2. Fuzzy inference system
3. Output membership functions



FIS Name: Untitled

FIS Type: mamdani

And method: min

Or method: max

Implication: min

Aggregation: max

Defuzzification: centroid

Current Variable

Name

Type

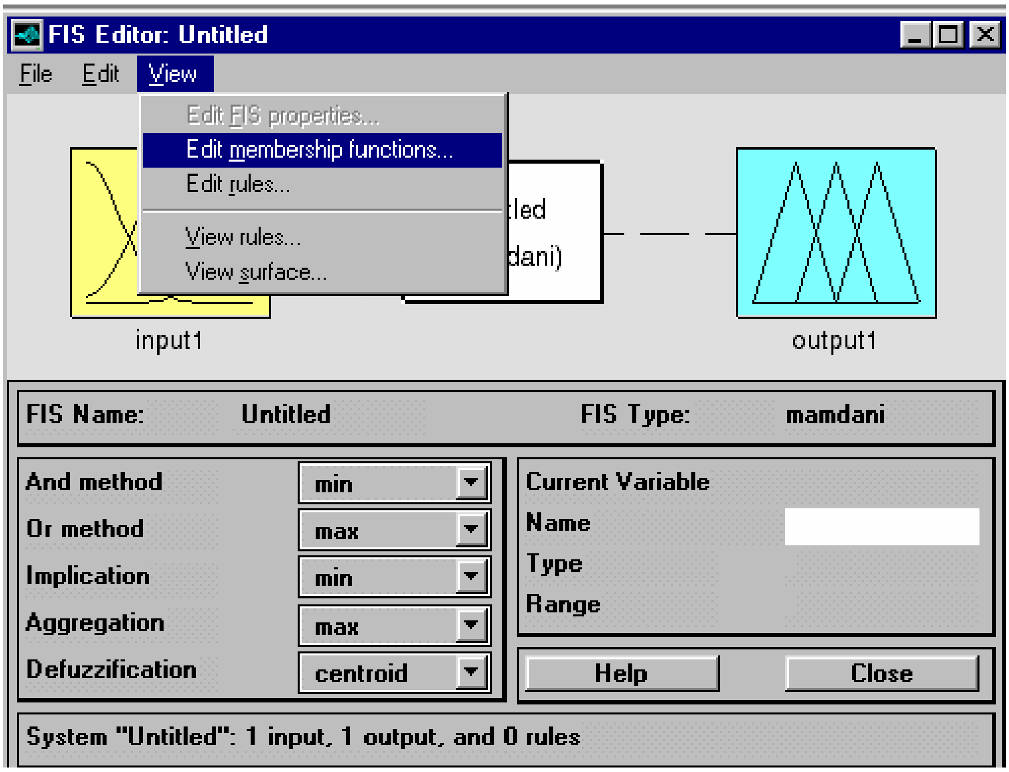
Range

System "Untitled": 1 input, 1 output, and 0 rules

Fig.2.2. GUI of Fuzzy toolbox. Study the content of GUI to have an overall understanding before proceeding.

Next activate input membership block by moving mouse on top of it and by clicking once. Activation is shown by a *red boundary* on the block. If you click twice, the **MEMBERSHIP FUNCTION EDITOR** opens up. This is needed when defining membership functions.

Another way to do the same is to use ***View*** menu (cf. Figure below). Input block is again activated first.



FIS Name: Untitled

FIS Type: mamdani

And method: min

Or method: max

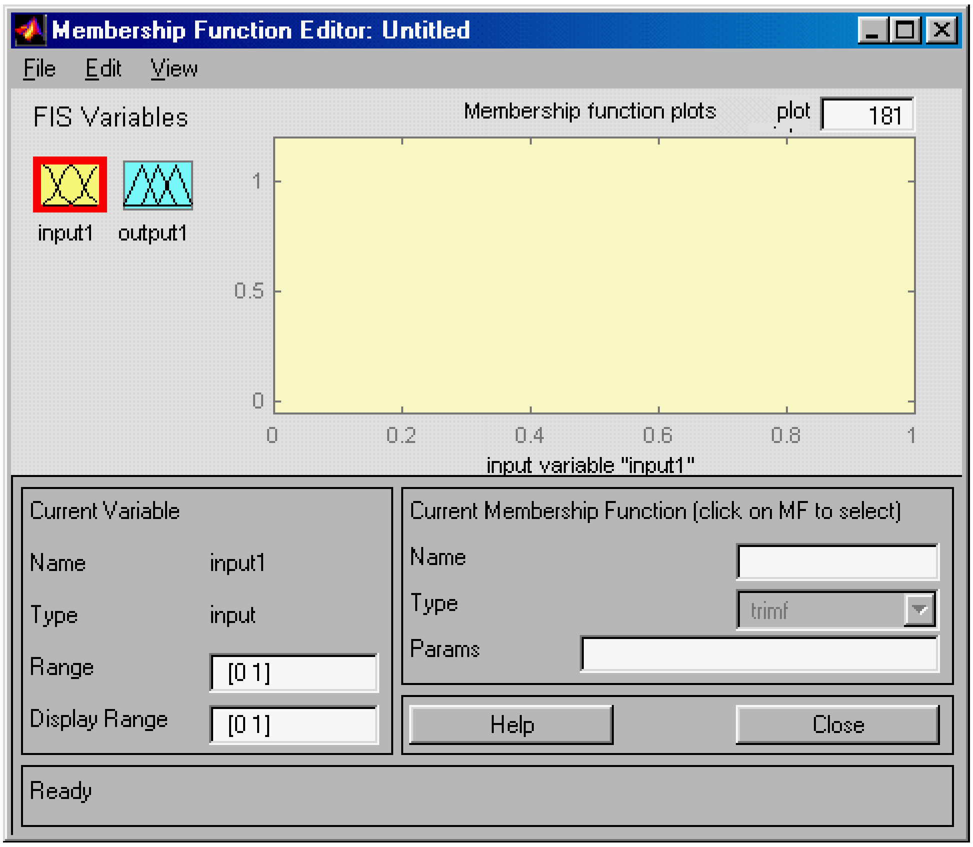
Implication: min

Aggregation: max

Defuzzification: centroid

System "Untitled": 1 input, 1 output, and 0 rules

Fig.2.3. Editing membership functions from ***View*** menu.



plot 181

Current Variable

Name input1

Type input

Range [0 1]

Display Range [0 1]

Current Membership Function (click on MF to select)

Name

Type trimf

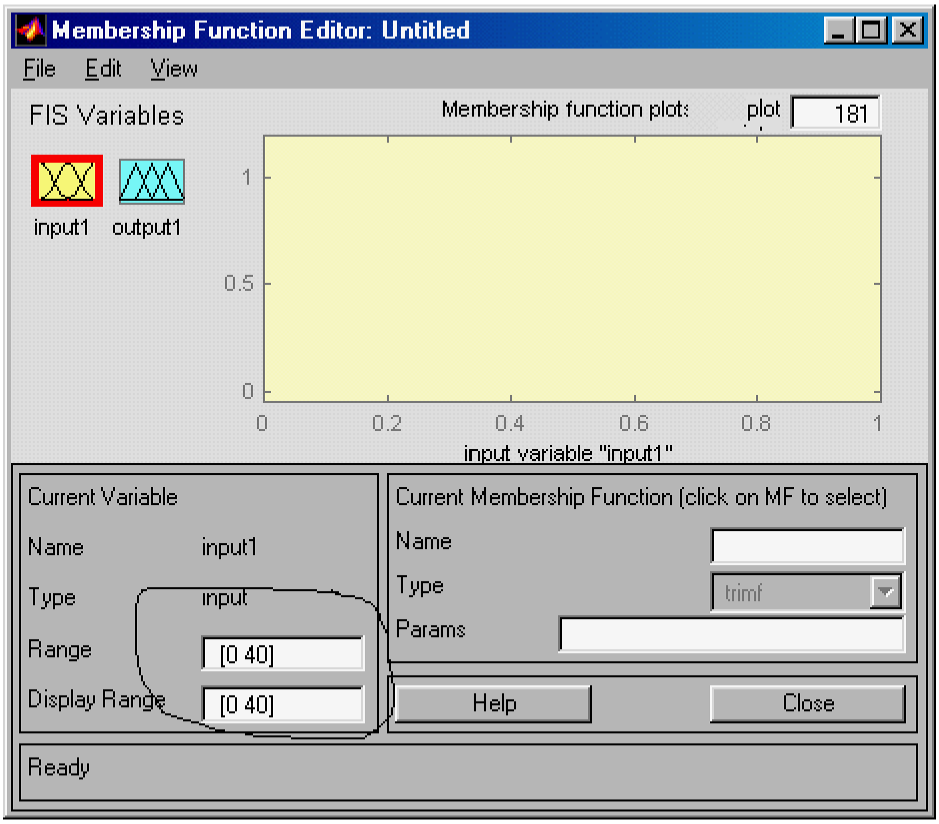
Params

Ready

Fig.2.4. Display to edit input membership functions.

First change *Range* to the one given in the problem or to [0^0,40^0]. This is done by moving the cursor to the *Range* area and clicking once. Then you can correct the figures in the range domain as you would correct text in a text document. Note that you have to leave space between the numbers.

Next click *Close* or anywhere in the light grey area above with the mouse. The result is shown below.



plot 181

Current Variable

Name input1

Type input

Range [0 40]

Display Range [0 40]

Current Membership Function (click on MF to select)

Name

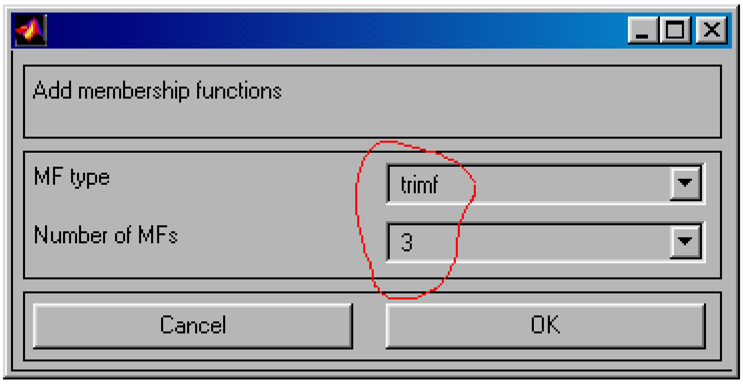
Type trimf

Params

Ready

Fig.2.5. Change the range of input variable to [0^0,40^0].

Now you are ready to define new membership functions. Choose ***Add MF's*** from EDIT menu. The following display is shown.



Add membership functions

MF type trimf

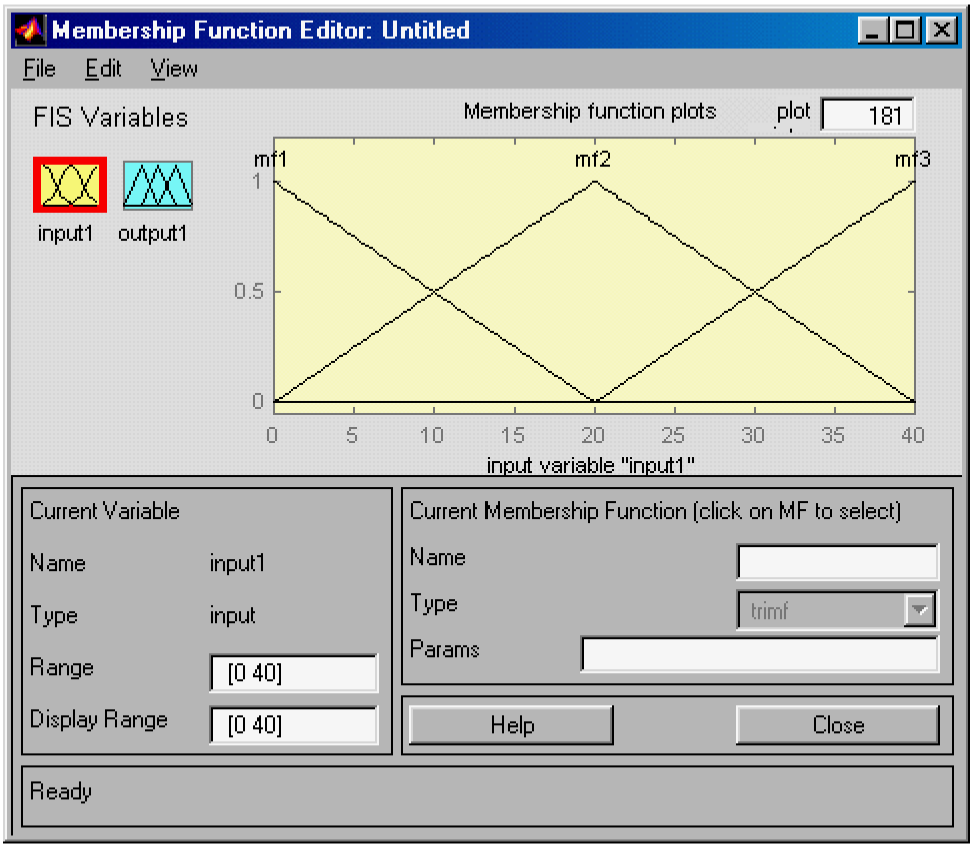
Number of MFs 3

Fig.2.6. Type and number of membership functions can be chosen.

Default value is three triangular (*trimf*) (3) membership functions as seen on the display. These will divide the range into three equal parts.

Click OK, if you are happy with the default values, otherwise make your own choice. (Try e.g. five *gaussian* membership functions. Different membership functions can be found under *MF type*.)

The result is shown below. Note that the default names for the membership functions are *mf1, mf2* and *mf3.*



plot 181

Current Variable

Name input1

Type input

Range [0 40]

Display Range [0 40]

Current Membership Function (click on MF to select)

Name

Type trimf

Params

Ready

Fig.2.7. The result of choosing three, triangular membership functions, which by default are named *mf1, mf2* and *mf3.*

2.3. Fuzzy set operations

Let *A* and *B* be two fuzzy sets in universal discourse *U*. Their corresponding membership functions are

μ_A **and** μ_B.

The basic set operations *union, intersection* and *complement* are defined by the membership functions as follows:

Union

$$\mu_{A\cup B}(u) = \max\{\mu_A(u), \mu_B(u)\}, \quad \forall\, u \in U$$

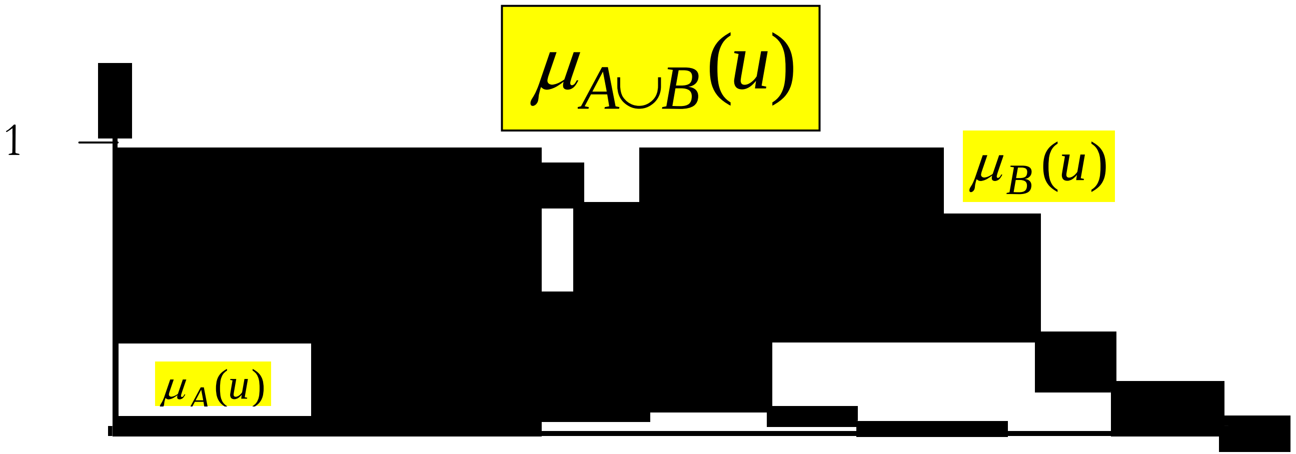


Fig.2. 8. The membership function of **union** $A \cup B$ is defined by taking the **maximum** of membership functions of *A* and *B*.

Intersection

$$\mu_{A\cap B}(u) = \min\{\mu_A(u), \mu_B(u)\}, \quad \forall\, u \in U$$

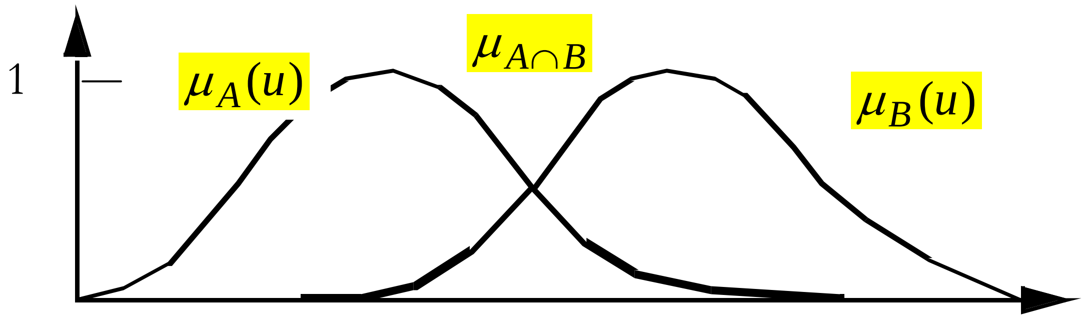


Fig.2.9. *The membership function of* ***intersection*** *of fuzzy sets* $A \cap B$ *is defined by taking the* ***minimum*** *of membership functions of A* ***or*** *B.*

Complement

The membership function of the ***complement*** of a fuzzy set *A*, $\overline{A}$ is defined as follows

$$\mu_{\overline{A}}(u) = 1 - \mu_A(u), \quad \forall\, u \in U$$

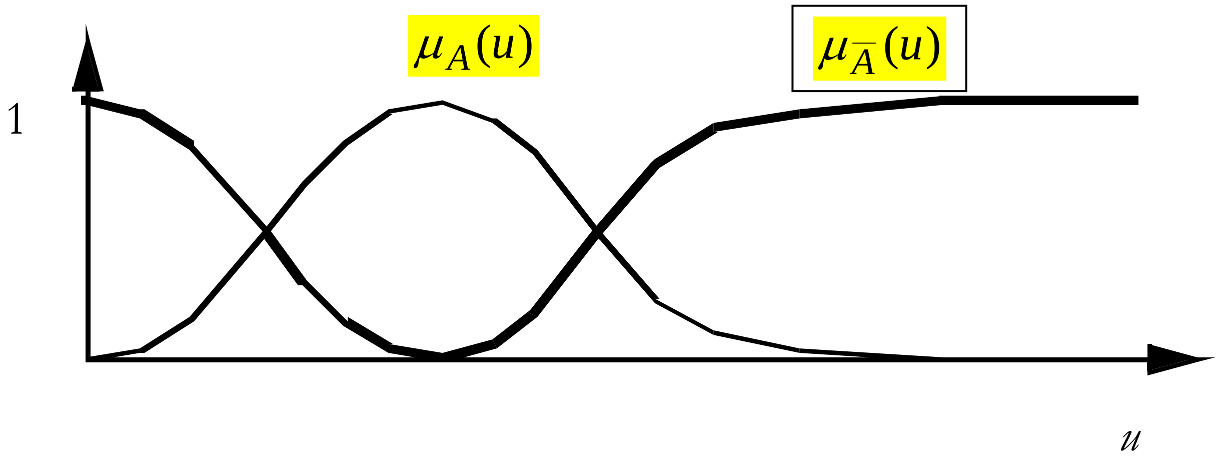


Fig.2.10. Membership functions of a fuzzy set *A* and its complement $\overline{A}$.

2.3. Fuzzy logical operations

The membership function of the logical operation ***or*** of two fuzzy sets *A* and *B* is

$$\mu_{A\vee B}(u,v)=\max\{\mu_A(u),\mu_B(v)\},\quad u\in U\wedge v\in V$$

The membership function of the logical operation ***and*** of two fuzzy sets *A* and *B* is

$$\mu_{A\wedge B}(u,v)=\min\{\mu_A(u),\mu_B(v)\},\quad u\in U\wedge v\in V$$

The membership function of the logical operation ***not*** of a fuzzy set *A* is

$$\mu_{\overline{A}}(u)=1-\mu_A(u),\quad u\in U$$

EXAMPLE

RULEBASE

Rule *1*: If x is A_1 and y is B_1 then z is C_1,
Rule *2*: If x is A_2 and y is B_2 then z is C_2.

Here A_1, B_1 and C_1, A_2, B_2 and C_2 are fuzzy sets.

Interpretation:

1.	**and**	**corresponds to**	**min**
2.	**or**	**corresponds to**	**max**
3.	**then**	**corresponds to**	**min**

MORE GENERALLY

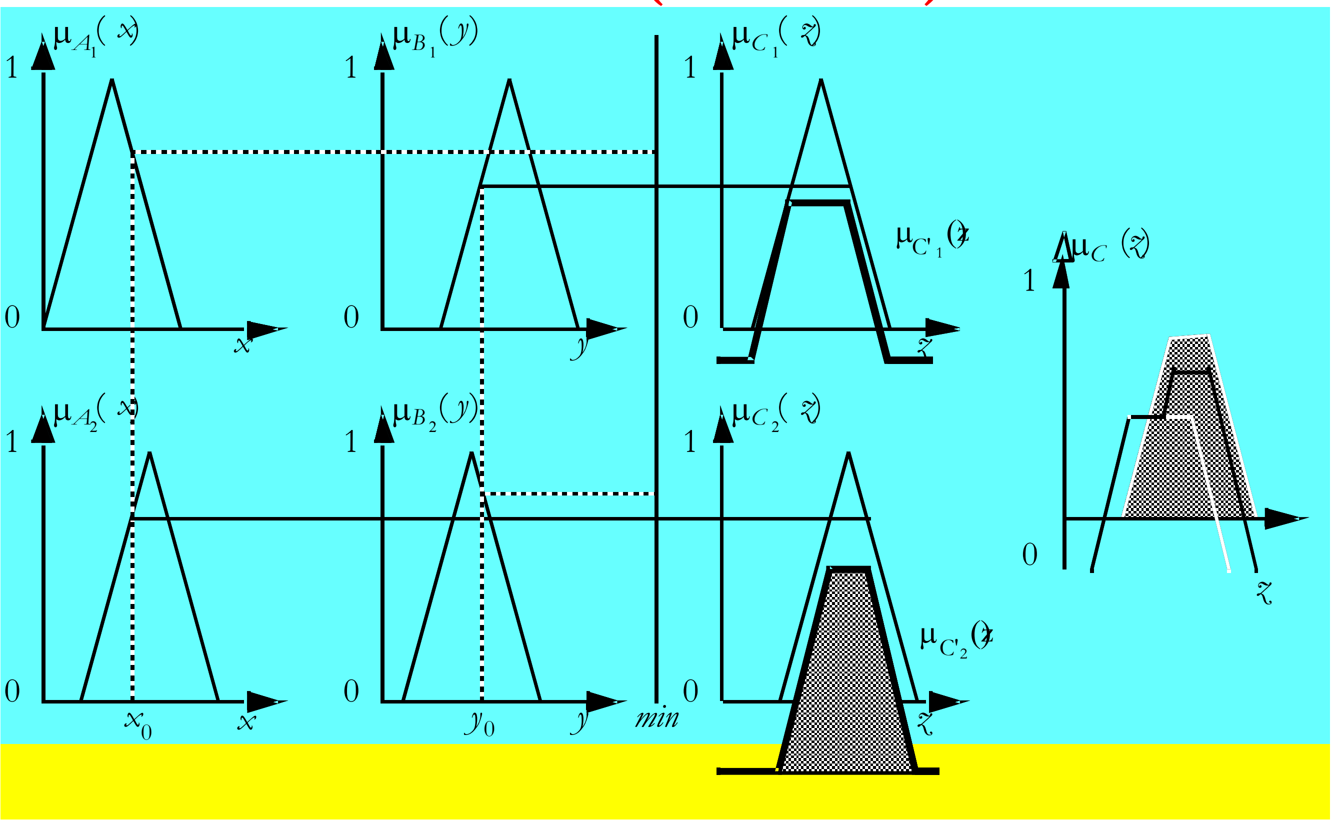
1. $\alpha_i = T(\mu_{A_i}(x_0), \mu_{B_i}(y_0))$

2. $\mu_{C_i'}(z) = T(\alpha_i, \mu_{C_i}(z))$

To combine the rules

3. $\mu_C(z) = T^*(\mu_{C_1}(z), \mu_{C_2}(z), \ldots, \mu_{C_N}(z))$

FUZZY INFERENCE (AND-OR)



Evaluate the first rule

1. $\alpha_1 = \min(\mu_{A_1}(x_0), \mu_{B_1}(y_0))$
2. $\mu_{C_1'}(z) = \min(\alpha_1, \mu_{C_1}(z))$

Evaluate the second rule

1. $\alpha_2 = \min(\mu_{A_2}(x_0), \mu_{B_2}(y_0))$
2. $\mu_{C_2'}(z) = \min(\alpha_2, \mu_{C_2}(z))$

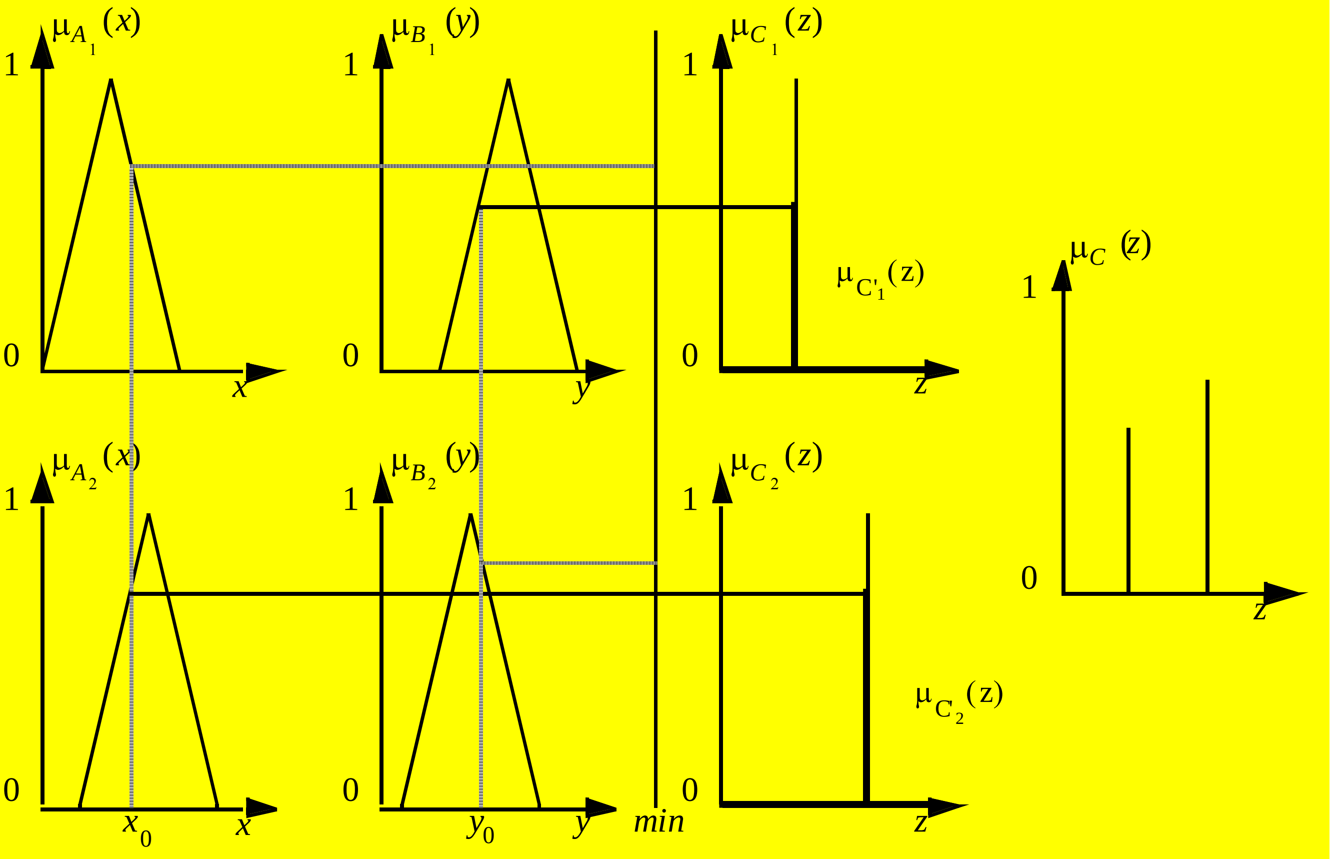
Combine both rules (Aggregate)

$$\mu_{C'}(z) = \max(\mu_{C_1'}(z), \mu_{C_2'}(z))$$

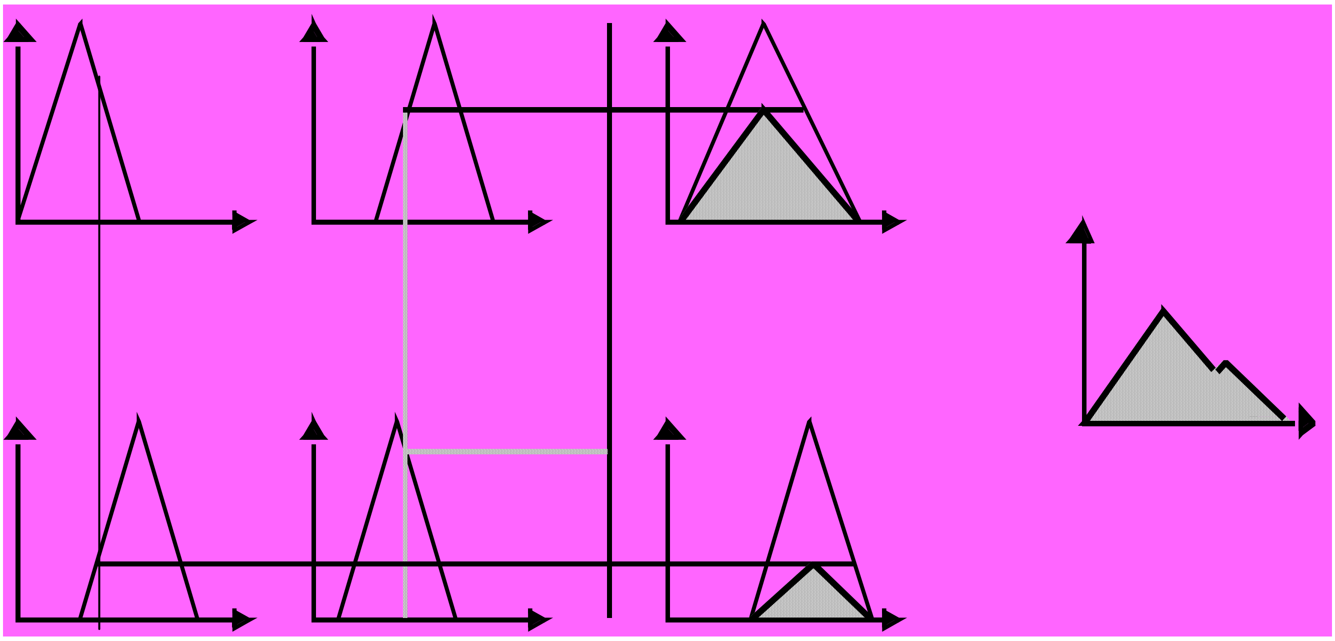
Defuzzify

Compute the center of gravity

OUTPUT MEMBERSHIP FUNCTIONS SINGLETONS



FUZZY INFERENCE (PRODUCT)



2.3.1. T-norm

Triangular norm

Mapping $T:[0,1]\times[0,1]\rightarrow[0,1]$ is called **T-norm if**

it satisfies the following criteria:

$T(a,b)=T(b,a)$ commutativity

$T(T(a,b),c)=T(a,T(b,c))$ associativity

$a\leq c\wedge b\leq d=T(a,b)\leq T(c,d)$ nondecreasing

$T(a,1)=T(1,a)=a$, $T(0,0)=0$ boundary

Example: *Intersection of sets*.

REMARK: In fuzzy control *T-norm* is used to connect different propositions in connection of ***and***-operation. The arguments are then the corresponding membership functions.

The most common T-norms are

1. Minimum $\min(a,b)$

2. Product ba

2.3.2. T-conorm

Triangular conorm

Mapping $T^{*}:[0,1]\times[0,1]\rightarrow[0,1]$ is called **T-conorm,**

if it is T-norm and in addition satisfies

$$T*(a,0)=T*(0,a)=a\,,T*(1,1)=1$$

Example 2.2:

T-conorm is generally used to connect propositions in connection of *or*-operation.

The most common T-conorms:

1. Maximum $\max(a,b)$
2. In probability theory *a+b-ab*
3. For triangular membership functions sum (*a+b*) is used

   (although it does not satisfy the above conditions)

2.4. Fuzzy reasoning

1. **Generalized Modus Ponens (GMP)**

   (forward chaining)

2. **Generalized Modus Tollens (GMT)**

   (backward chaining)

Example 2.3: Consider two fuzzy sets A and B, and their complements $\overline{A}$ and $\overline{B}$. Then

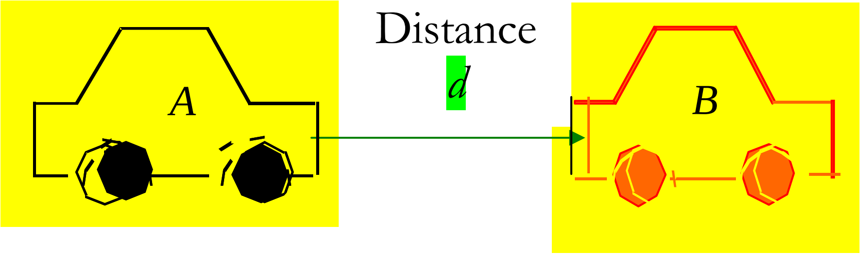
(GMP)

Fact 1:	x is $\overline{A}$
Premise 2:	If x is A then y is B
Conclusion:	y is $\overline{B}$

(GMT)

Fact 1:	y is $\overline{B}$
Premise 2:	If x is A then y is B
Conclusion:	x is $\overline{A}$

Example 2.4: *Car driving*



SAFE
DISTANCE

Fig.2.11. Cars A and B on a highway.

You are driving car A on a highway. You want to keep a ***safe distance*** to car B in front of you. Design a (simplified) fuzzy-logic system, which satisfies the requirements.

Proceed as follows:

a. Determine the required fuzzy variables (input/output) and their ranges.

b. Form the rule base.

c. Use fuzzy reasoning to check the operability of the
rulebase.

SOLUTION:

a. **Fuzzy variables**. Start with a simple case.

INPUT: Distance *d*

OUTPUT: Breaking power *b* (gas pedal)

Three (3) membership functions are chosen for both input and output.

Membership functions for INPUT:

Distance *d* (meters): ***short, medium, long.***

Membership functions for OUTPUT:

Breaking power *b* (%): ***large, medium, none***

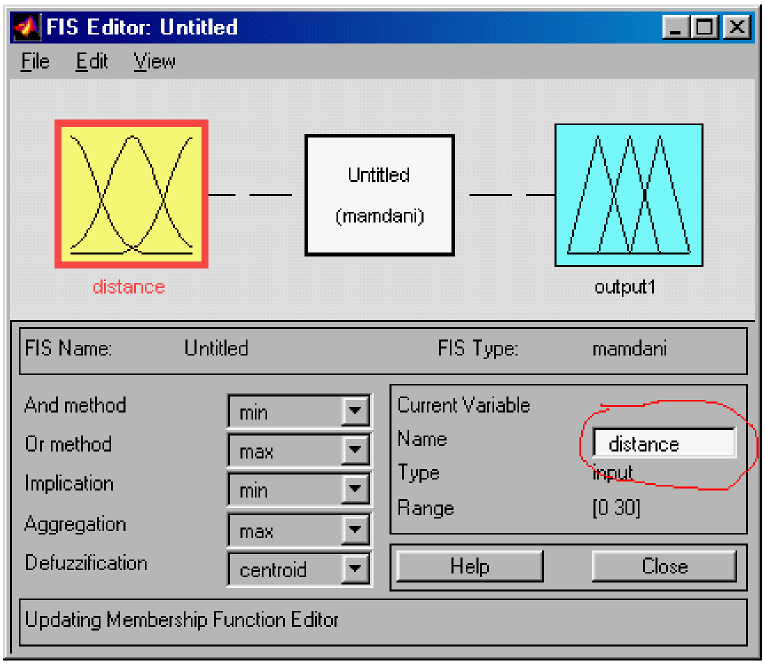
We will use the fuzzy toolbox to define the fuzzy system by giving numerical values for the variables indicated.

In MATLAB type

» fuzzy

This opens the GUI. Activate the input window.

Give a name to the fuzzy input variable. Call it *distance*. Click *Close*.



FIS Name: Untitled

FIS Type: mamdani

And method: min

Or method: max

Implication: min

Aggregation: max

Defuzzification: centroid

Current Variable

Name: distance

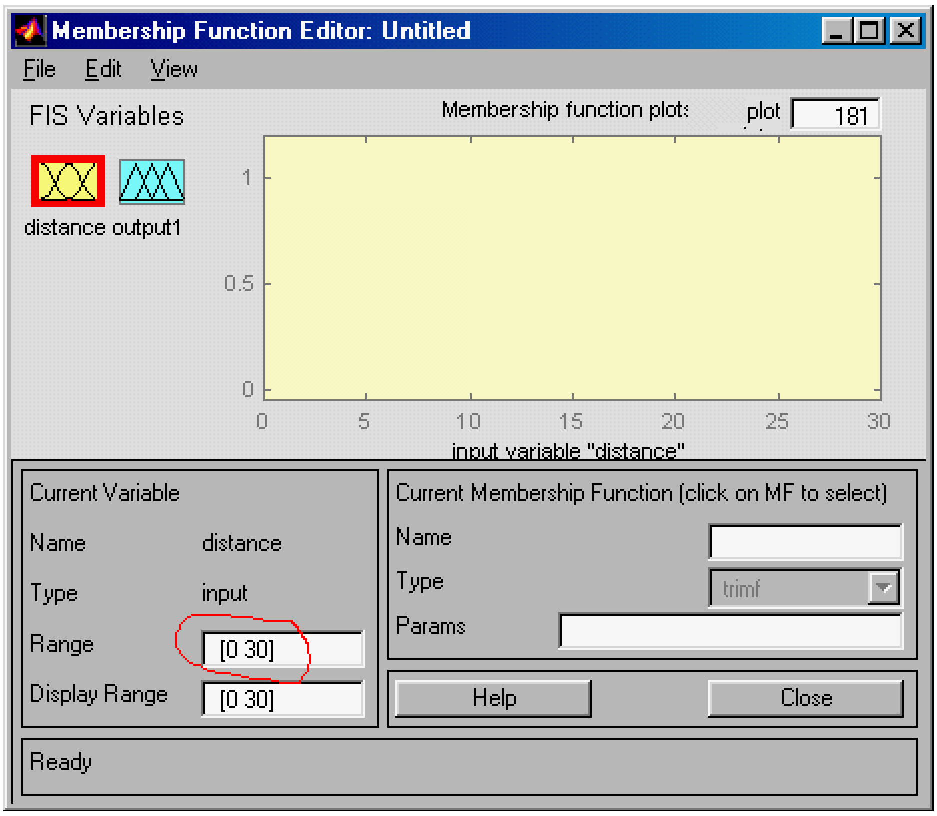
Type: input

Range: [0 30]

Updating Membership Function Editor

Fig.2.12. Naming the input variable in GUI.

Next click the input block twice with mouse to open the membership function window. First define the range, say from 0 to 30 m/s.


Membership Function Editor: Untitled
File
Edit
View
FIS Variables
distance
output1
Membership function plots
plot
181
1
0.5
0
0
5
10
15
20
25
30
input variable "distance"
Help
Close

Current Variable

Name distance

Type input

Range [0 30]

Display Range [0 30]

Current Membership Function (click on MF to select)

Name

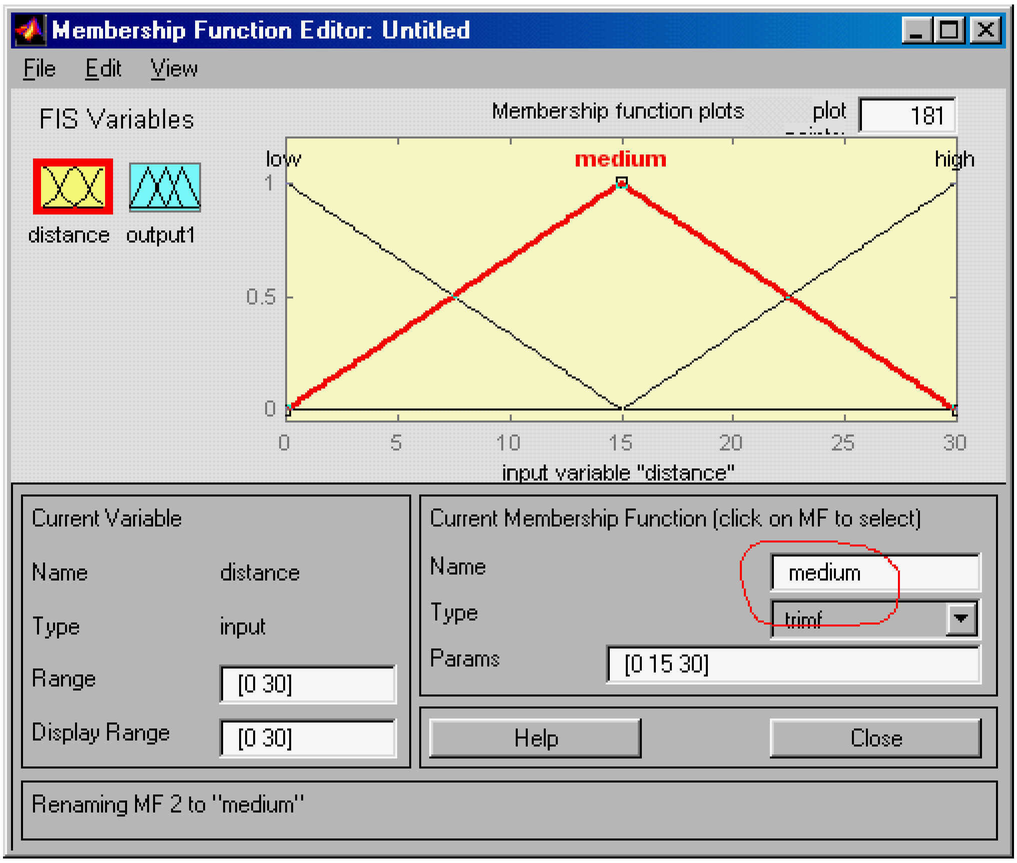
Type trimf

Params

Ready

Fig. 2.13. Set the range in GUI.

Next choose from Edit ***Add MFs***. Pick the default values: 3 triangular membership functions. Give a name to each: Call them *high*, *medium*, and *short*. When you are finished, click *Close*.



Current Variable

Name distance

Type input

Range [0 30]

Display Range [0 30]

Current Membership Function (click on MF to select)

Name medium

Type trimf

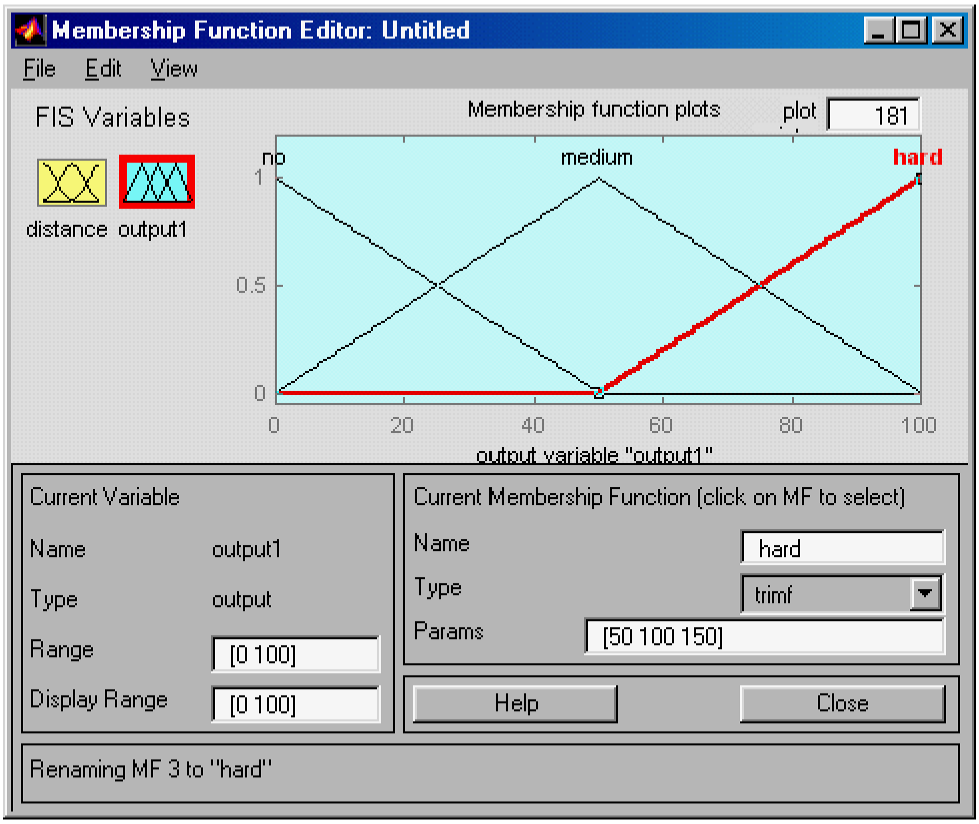
Params [0 15 30]

Renaming MF 2 to "medium"

Fig. 2.14. Three triangular membership functions have been chosen for the input variable *distance*. The middle one has been activated and renamed as *medium*.

Repeat the same procedure with the output *b*, breaking power.

Define the name of the output, break, and its range. Use three membership functions: ***hard, medium*** and ***no.*** The following GUI display is obtained.



Membership Function Editor: Untitled

Current Variable

Name output1

Type output

Range [0 100]

Display Range [0 100]

Current Membership Function (click on MF to select)

Name hard

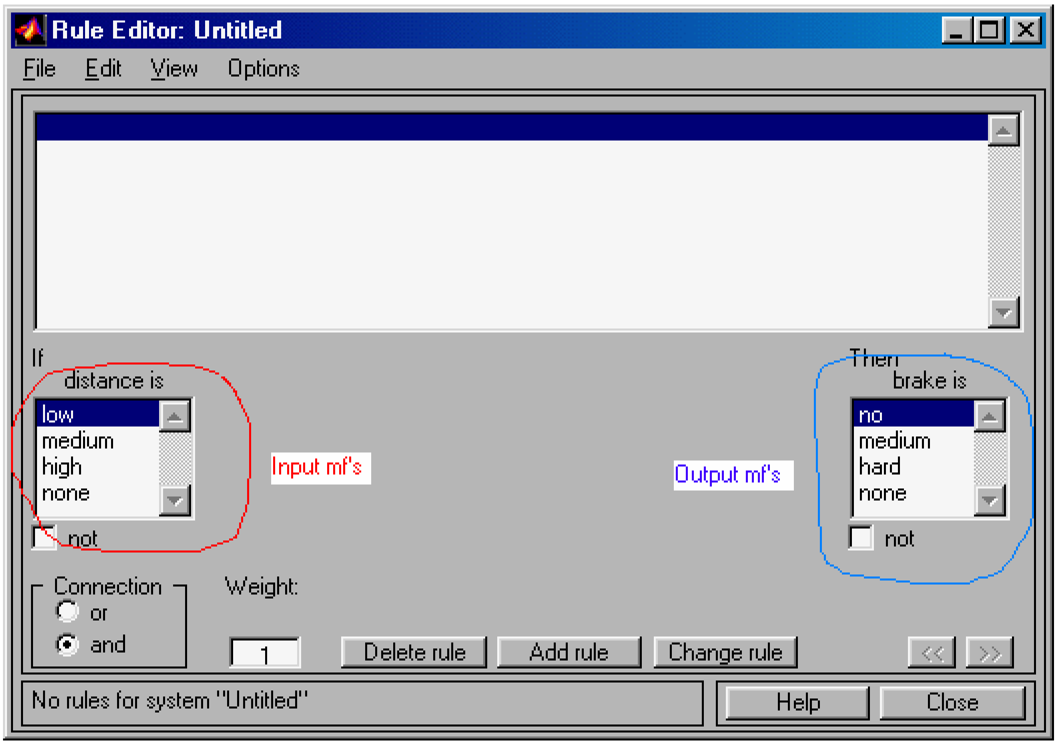
Type trimf

Params [50 100 150]

Renaming MF 3 to "hard"

Fig. 2.15. Three triangular membership functions are chosen for the output variable breaking. The last one has been activated and renamed as *hard.*

What is missing from the fuzzy system now is the rule base. Open *View* menu and click *Edit rules*. Then the following display opens.



No rules for system "Untitled"

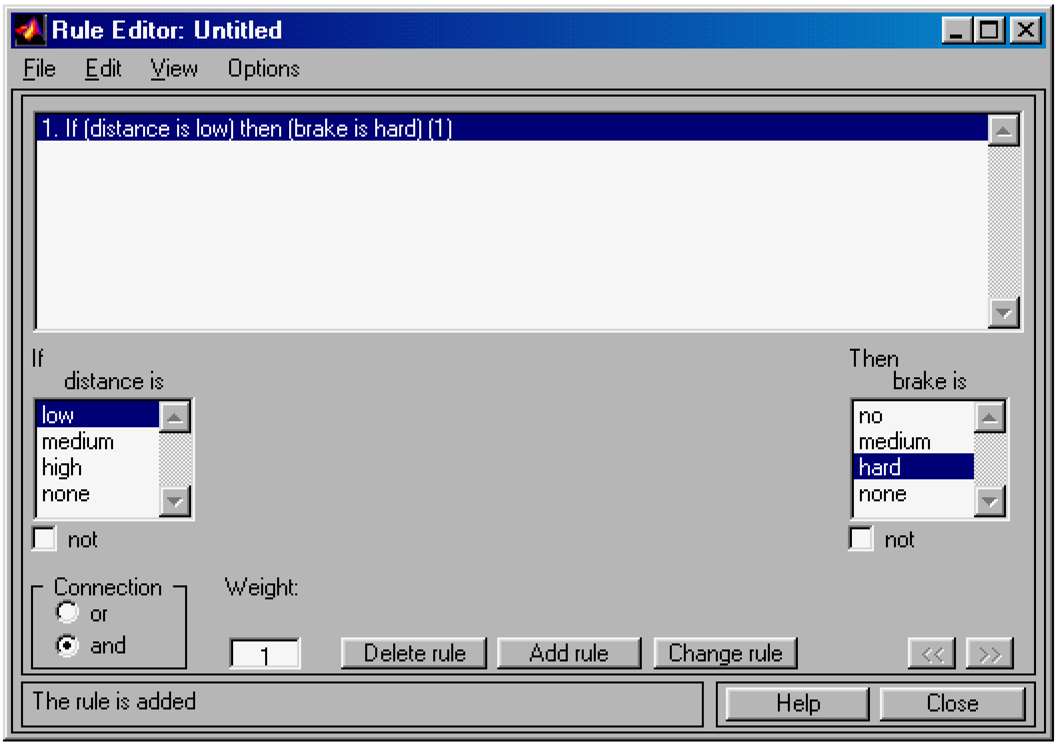
Fig. 2.16. Rule Editor display. On the right, the input side. On the left, the output side.

The left-hand side contains the membership functions of the input, *distance*. The right-hand side has the membership functions of the output, *brake*. If the input side has several variables, which are connected either by *and* or *or,* the *Connection* block is in the lower left-hand corner. In this case we only have one input variable, so the connective is not used. The weight factor (default value = 1), indicates the importance of the rule in question.

The construction of the rule base is the hardest part of the design task. Here a simple-minded rule base is constructed based on driving experience. Typical rule is

If distance is low, then brake is hard.

With mouse choose the membership function *low* for the distance and *hard* for brake. This is done in the figure above. Then click ***Add rule***. The result is seen below.



1. If (distance is low) then (brake is hard) (1)

If distance is: low (selected), medium, high, none; □ not

Then brake is: no, medium, hard (selected), none; □ not

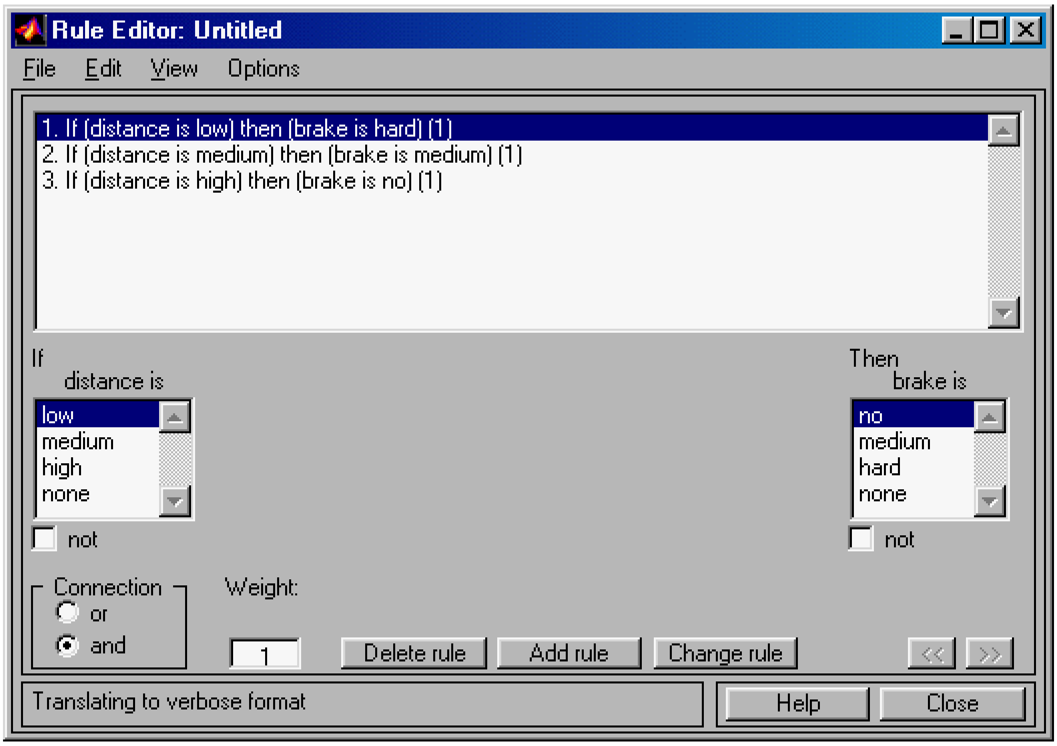
Connection: ○ or, ● and

Weight: 1

The rule is added

Fig. 2.17. Setting up a rule with Rule Editor.

Let us set two other rules, one for *medium* distance and the other for *long* distance. Our simple, rule base is now complete. Click *Close*.



1. If (distance is low) then (brake is hard) (1)
2. If (distance is medium) then (brake is medium) (1)
3. If (distance is high) then (brake is no) (1)

If
distance is
low
medium
high
none

□ not

Then
brake is
no
medium
hard
none

□ not

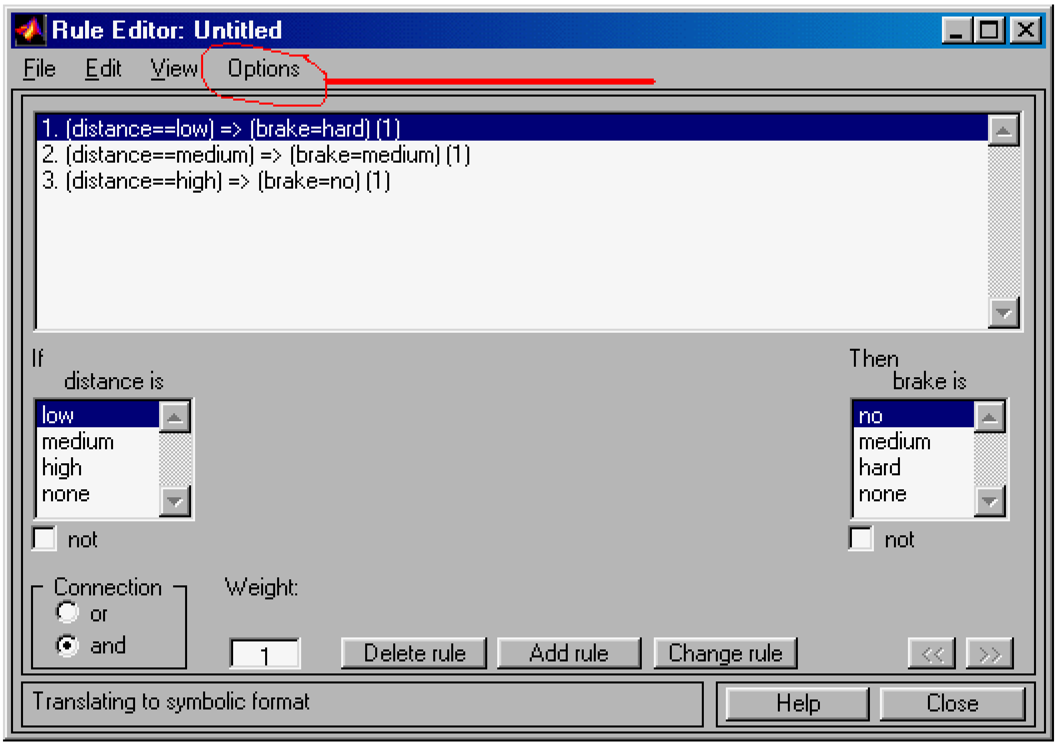
Connection
○ or
● and

Weight:
1

Translating to verbose format

Fig. 2.18. Complete rule base of three rules. Note the weighting parameter (1) at the end of each rule, i.e., all rules have the same weighting.

Now the design of the fuzzy system is complete. The Toolbox provides two more interesting ways expressing the rule base.



```
1. (distance==low) => (brake=hard) (1)
2. (distance==medium) => (brake=medium) (1)
3. (distance==high) => (brake=no) (1)
```

distance is: low, medium, high, none; □ not

brake is: no, medium, hard, none; □ not

Connection: ○ or ● and

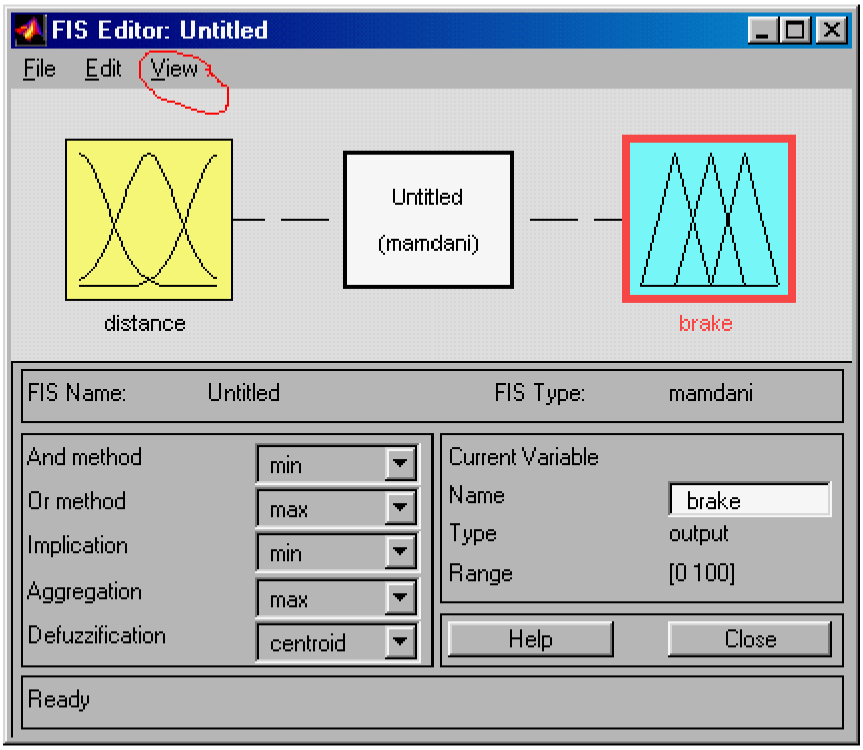
Weight: 1

Translating to symbolic format

Fig. 2.19. The rule base can be expressed in two other ways in GUI. Here verbose format is shown.

Check under *Options* and there choose *Format*. Under that you can see that the rules as shown are given *verbose*. The other forms are *symbolic* and *indexed*. Check in what form the rule base is given in each case.

Viewing rules gives you the overall picture of the developed fuzzy system. From the main FIS editor, choose from *View* menu *View rules*.



FIS Name: Untitled

FIS Type: mamdani

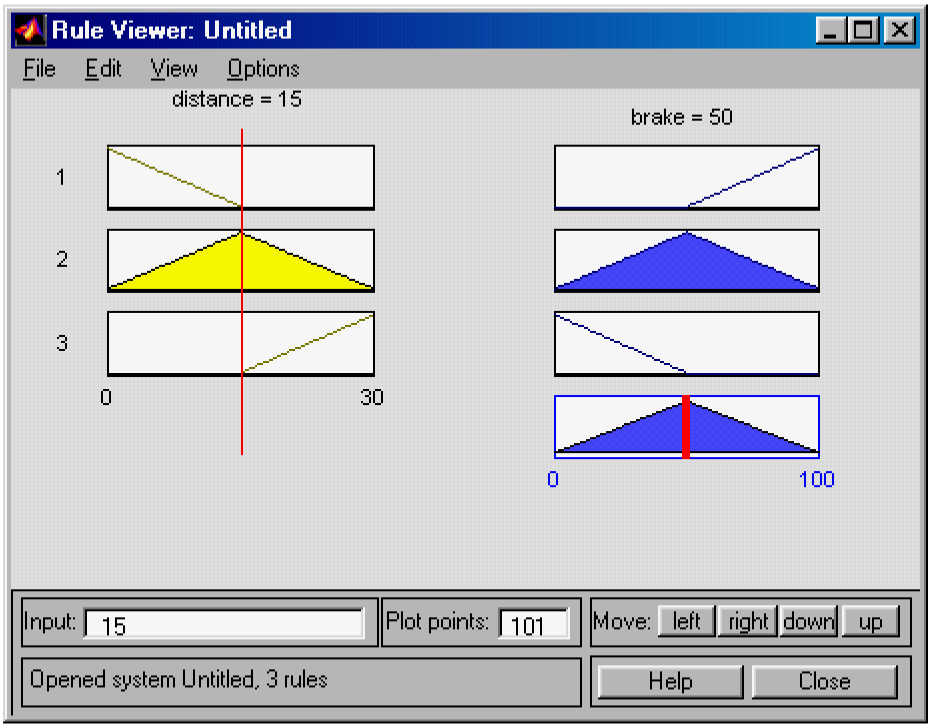
And method	min
Or method	max
Implication	min
Aggregation	max
Defuzzification	centroid

Current Variable

Name	brake
Type	output
Range	[0 100]

Fig. 2.20. By choosing *View* the rule base can be viewed differently.

The display *View rules* is shown below.



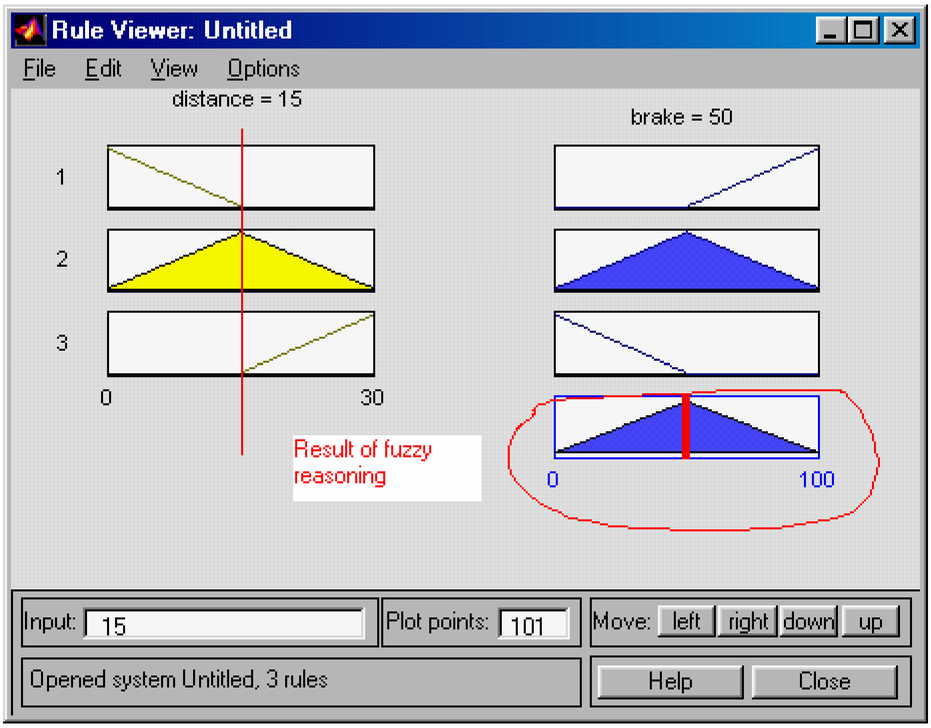
Input: 15

Plot points: 101

Opened system Untitled, 3 rules

Fig. 2.21. The rule base can be displayed graphically. The red line on the left indicates the value of the input, 15 m. Similarly the bar on the right hand side, indicates the output value, 50%.

On the left-hand side you can see the input, *distance* side and on the left the output, *brake* side. There are three rules and the corresponding triangular membership functions displayed. In the right-hand side lower corner is the result of fuzzy reasoning. At this point it is a fuzzy set. Applying defuzzification method, in the figure center of gravity has been chosen, a crisp value is obtained.



Result of fuzzy reasoning

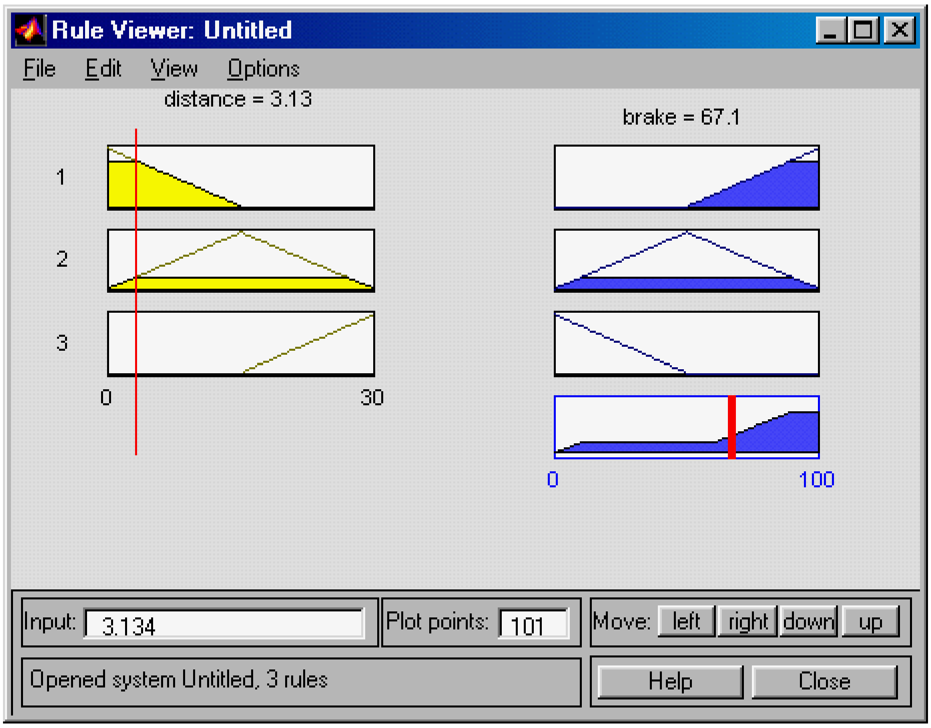
Input: 15

Plot points: 101

Opened system Untitled, 3 rules

Fig. 2.22. Result of fuzzy reasoning is brake = 50%.

Different input values can be tried by moving the red, vertical line on the left-hand side (Fig. 2.23).



distance = 3.13

brake = 67.1

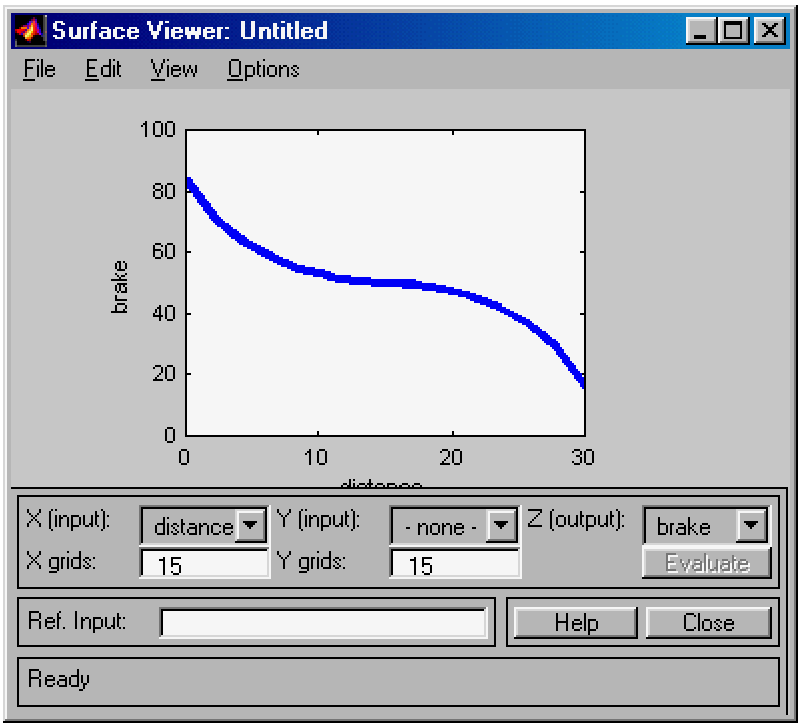
Input: 3.134

Plot points: 101

Opened system Untitled, 3 rules

Fig. 2.23. Changing the input value results in different output values.

Finally, the input-output mapping can be observed by viewing surface. Choose *View* menu and under it *View surface*. It is clear that our map is nonlinear. This is where the power of fuzzy systems is strong (Fig. 2.24).



X (input): distance

Y (input): - none -

Z (output): brake

X grids: 15

Y grids: 15

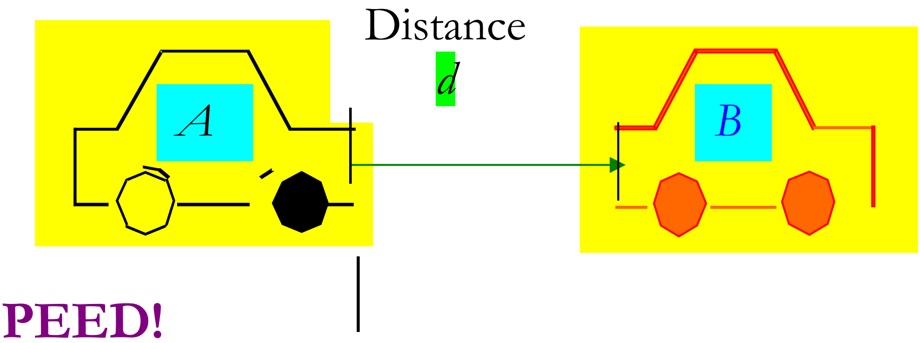
Ref. Input:

Ready

Fig. 2.24. The fuzzy system viewed as input-output mapping.

EXAMPLE 2.5: Car Driving - continues

What is missing in Example 2.4?



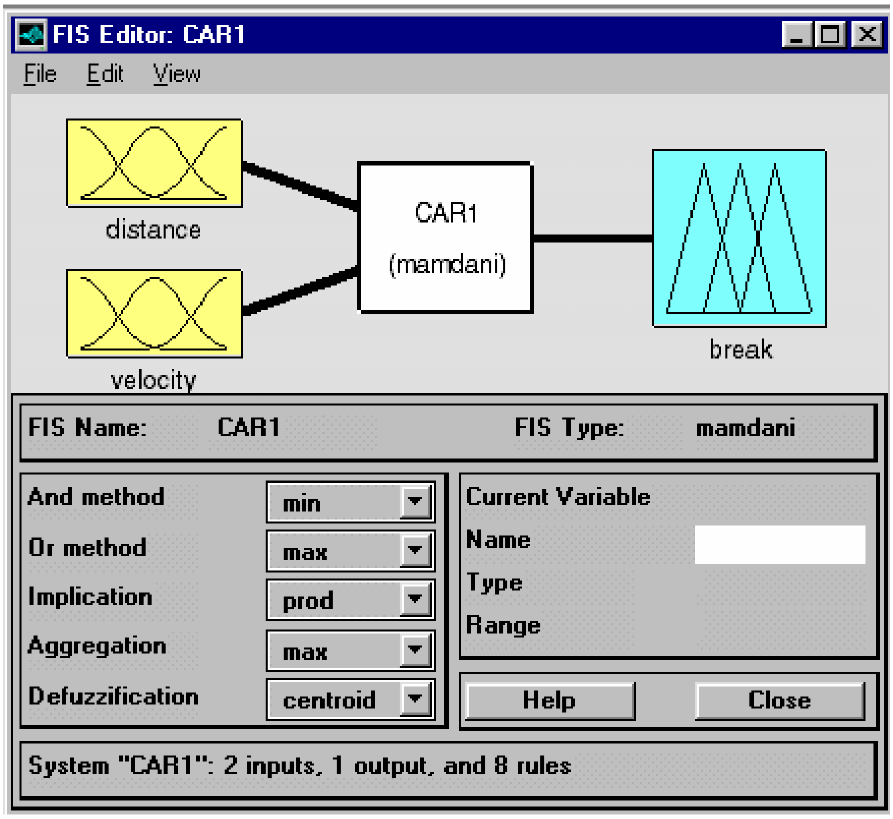
SPEED!

(relative speed between vehicles A and B).

SOLUTION:

Set up a new rule base with two inputs distance and speed, one output, braking power.

Here only the main steps are shown.



FIS Editor: CAR1

FIS Name: CAR1

FIS Type: mamdani

And method min

Or method max

Implication prod

Aggregation max

Defuzzification centroid

Current Variable

Name

Type

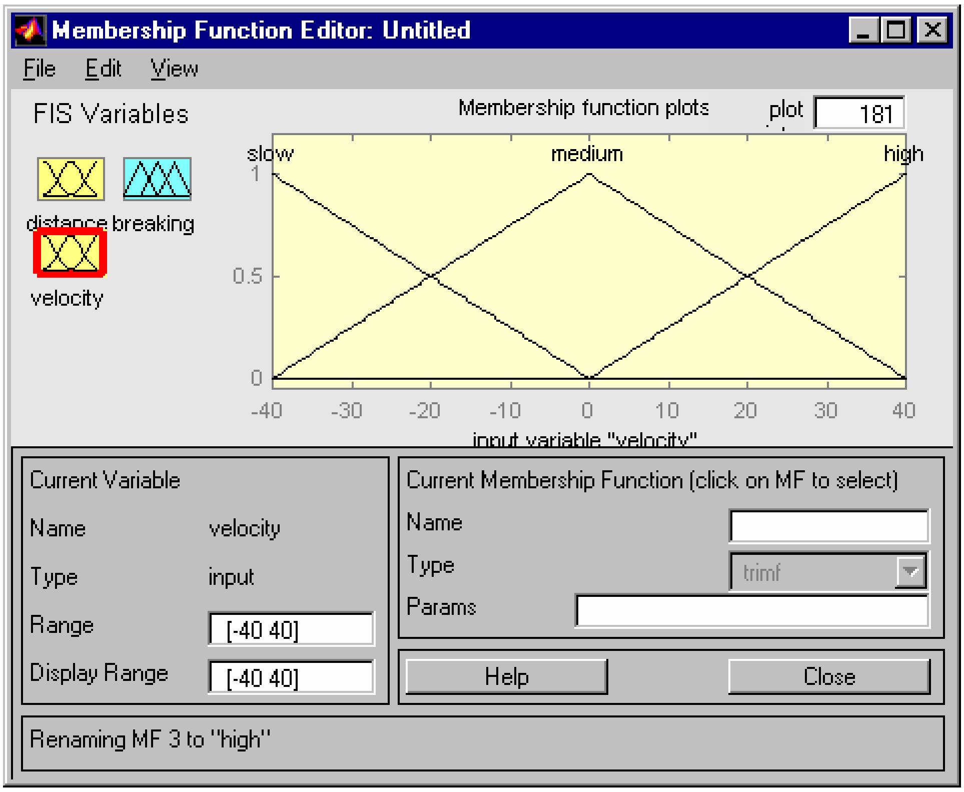
Range

System "CAR1": 2 inputs, 1 output, and 8 rules

Fig. 2.25. Fuzzy system with two inputs, distance and velocity, and one output, break.

The second input can be added by opening *Edit* menu. Otherwise the steps are as before. Remember to give a name for the added input. Call it *velocity*.

Only *velocity* membership functions are shown because they are new. Note that the speed range is from -40 to 40 km/h and the range has been divided into three membership functions: *slow, medium*, and *high* speed.



Current Variable

Name velocity

Type input

Range [-40 40]

Display Range [-40 40]

Current Membership Function (click on MF to select)

Name

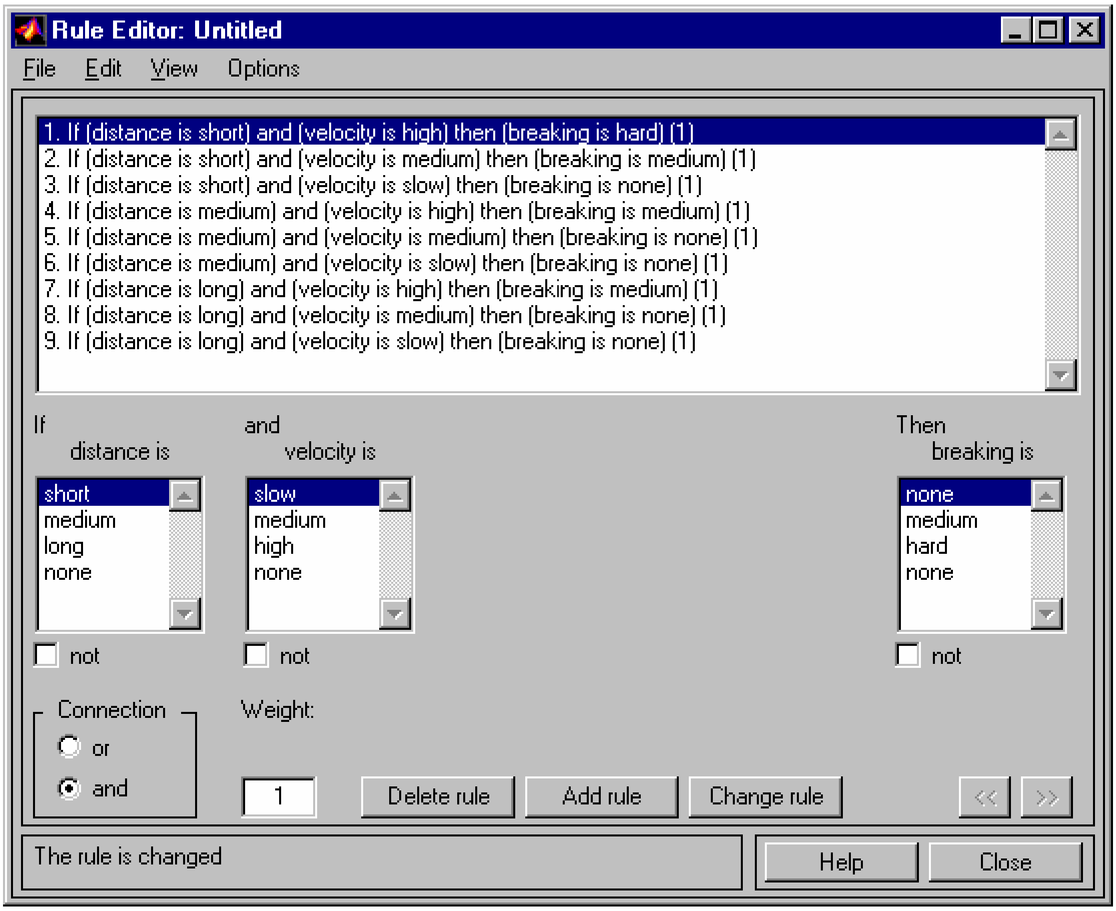
Type trimf

Params

Renaming MF 3 to "high"

Fig. 2.26. Membership functions for the new input, velocity.

Eight rules have been constructed, which are easy to understand (Fig. 2.27). Of course, the rules are not unique. Other rules could be used as well. The rule base must be viewed and tested to see its effectiveness and how well the system specifications are satisfied.



```
1. If (distance is short) and (velocity is high) then (breaking is hard) (1)
2. If (distance is short) and (velocity is medium) then (breaking is medium) (1)
3. If (distance is short) and (velocity is slow) then (breaking is none) (1)
4. If (distance is medium) and (velocity is high) then (breaking is medium) (1)
5. If (distance is medium) and (velocity is medium) then (breaking is none) (1)
6. If (distance is medium) and (velocity is slow) then (breaking is none) (1)
7. If (distance is long) and (velocity is high) then (breaking is medium) (1)
8. If (distance is long) and (velocity is medium) then (breaking is none) (1)
9. If (distance is long) and (velocity is slow) then (breaking is none) (1)
```

If distance is: short (selected), medium, long, none — □ not

and velocity is: slow (selected), medium, high, none — □ not

Then breaking is: none (selected), medium, hard, none — □ not

Connection: ○ or ● and

Weight: 1

The rule is changed

Fig. 2.27. A rule base for Example 2.5.

Viewing the rules is shown in Fig. 2.28.

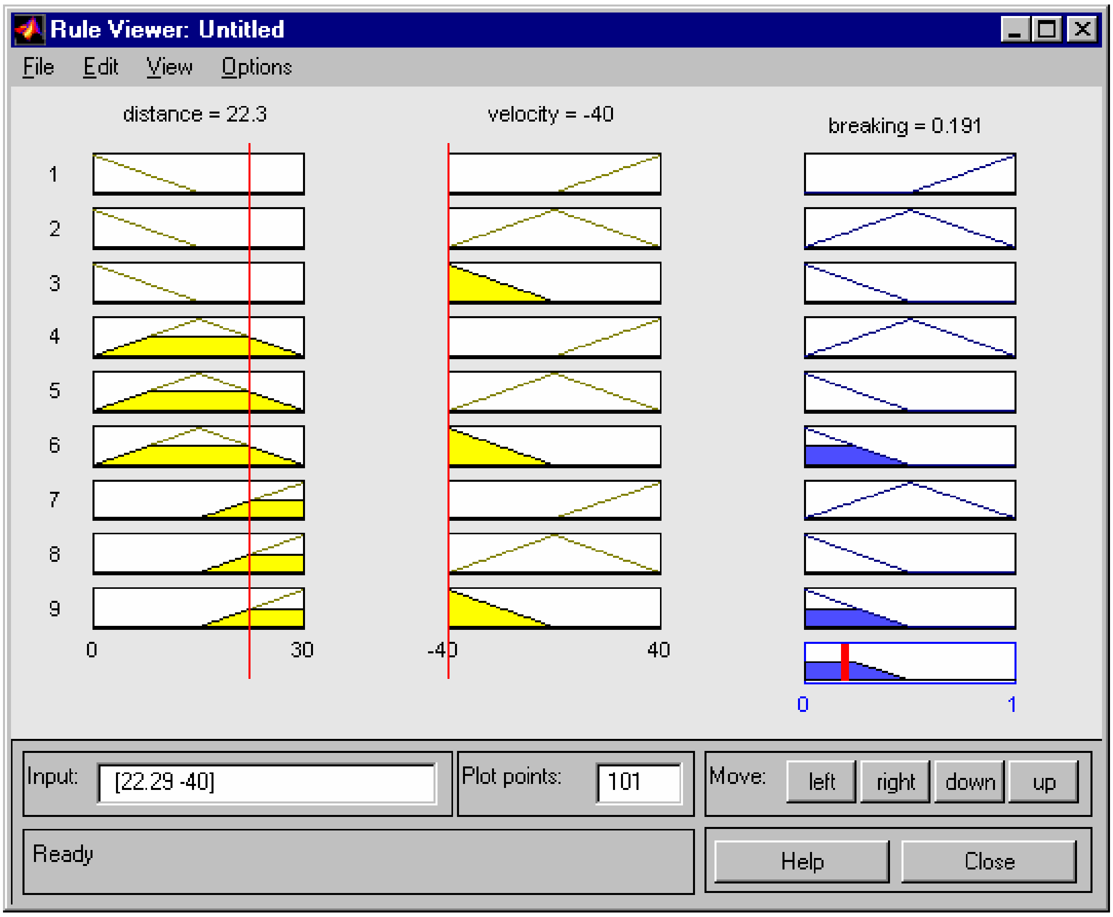
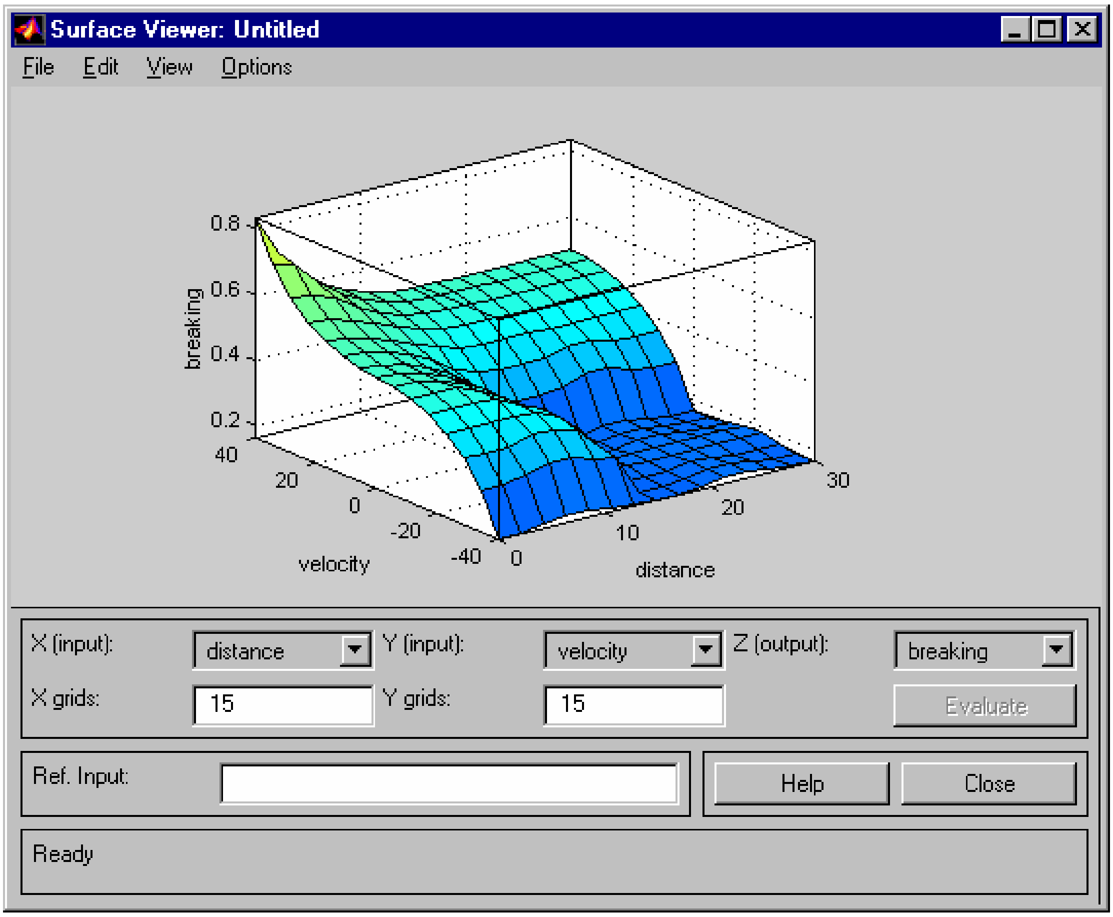


Fig. 2.28. The rule base is viewed graphically. Different inputs can be chosen to see what the output will be.

Again you can test the system by choosing different values of inputs. If the result is not satisfactory, then refine the system. E.g. it seems that the area of short distance and high speed does not give strong enough braking response.

Finally, the surface view of the rules is shown.



X (input): distance

Y (input): velocity

Z (output): breaking

X grids: 15

Y grids: 15

Ref. Input:

Ready

Fig. 2.29. The surface view of the constructed rule base.

Note that the right hand-side corner is flat with value zero over a fairly large area. You can change that by introducing a new membership function ***little*** for breaking power. You also have to change the rule base. This is done below.